A Fully Probabilistic Extension of Event-B

Mohamed Amine Aouadhi, Benoit Delahaye, Arnaud Lanoix

A Fully Probabilistic Extension of Event-B

Mohamed Amine Aouadhi, Benoît Delahaye, and Arnaud Lanoix

University of Nantes / LINA UMR CNRS 6241

Abstract. Event-B is a state-based formalism that enables the correct-by-construction development of systems. Several works have focused on the extension of Event-B for the description of probabilistic systems. While these extensions of Event-B allow replacing some non-deterministic choices with probabilities, some sources of non-determinism are left untouched. In this paper, we propose a fully probabilistic extension of Event-B where all the non-deterministic choices are replaced with probabilistic ones. We present the syntax and the semantics of this extension and illustrate our approach on a classical case study.

1 Introduction

As systems become more and more complex, it becomes necessary to add new modelling features in order to take into account complex system properties such as reliability [20], responsiveness [8,19], continuous evolution, energy consumption etc. In particular, as soon as systems include randomised algorithms [15], probabilistic protocols [3] or potentially failing components, probabilistic models are needed in order to reason about them.
In this way, several research works have focused on the extension of Event-B to allow the expression of probabilistic information in Event-B models. Event-B [1] is a formal method used for discrete systems modelling. It is equipped with *Rodin* [2], an open toolset for modelling and proving systems. The development process in Event-B is based on refinement: systems are typically developed progressively using an ordered sequence of models, where each model contains more details than its predecessor.
In [14], Abrial *et al.* have summarised the difficulties of embedding probabilities into Event-B. This paper suggests that probabilities need to be introduced as a refinement of *non-determinism*. In Event-B, non-determinism occurs in several places such as the choice between enabled events in a given state, the choice of the parameters values in a given event, and the choice of the value given to a variable through some non-deterministic assignments. The ideal probabilistic extension of Event-B should thus allow using probabilities in all these places. Unfortunately, to the best of our knowledge, the existing works on extending Event-B with probabilities, which we recall in the next paragraphs, have focused on refining non-deterministic assignments into probabilistic assignments while leaving other sources of non-determinism untouched.

Qualitative probabilistic Event-B. In [11], Hallerstede *et al.* propose to express probabilistic properties in Event-B by focusing on a qualitative aspect of probability. In this proposition, non-deterministic assignments can be refined into *qualitative* probabilistic

assignments where the actual probability values are not specified. The Event-B semantics and proof obligations are then adapted to this new setting. In [21], the same authors study the refinement of qualitative probabilistic Event-B models and propose a tool support inside Rodin.

Quantitative probabilistic Event-B. Other works [16,18,17] have extended the approach presented above. The authors propose to refine non-deterministic assignments by *quantitative* probabilistic assignments where, unlike in [11], the actual probability values are specified. This new proposition is then exploited in order to assess several system properties such as reliability and responsiveness.

Although these extensions of Event-B allow refining the non-deterministic assignments into probabilistic ones, the other sources of non-determinism are left untouched. While we agree that non-determinism is a very important feature in system modelling, we consider that it is the responsibility of the developer to choose where to use non-determinism and where to refine it into probabilities. Our aim is thus to pursue the existing works by allowing the refinement of all other potential sources of non-determinism into probabilistic choices. In the long-term, we plan on developing a probabilistic extension of Event-B where probabilities can be introduced in each refinement step in order to refine some (but not necessarily all) non-deterministic choices. This global objective can be divided into many specific objectives. In particular, it is necessary to allow expressing probabilistic choices in the place of all potential non-deterministic choices in Event-B. Another important point is to derive, study and adapt the proof obligations specific to the probabilistic extension of Event-B. Finally, we will have to investigate the refinement process between non-deterministic and (partially) probabilistic Event-B models. Our global objective includes also studying some probabilistic properties that can be expressed and verified on (partially) probabilistic Event-B models. Obviously, this global objective is very ambitious and we can only achieve it by progressing on a step-by-step basis.

As a first step toward this long term goal, we present in this paper a *fully probabilistic* extension of Event-B where *all* non-deterministic choices are replaced with probabilistic ones. As in [16,18,17], we replace non-deterministic assignments with quantitative probabilistic assignments. Moreover, we propose to allow choosing the parameter values by using *uniform* probability distributions on their domain of acceptable valuations. Finally, we annotate all the events with *weights*, these weights are expressions over the variables of the models. As a consequence, the probability of choosing an event among others in a given state can evolve according to the model states. In order to prove the correctness of our approach, we express the semantics of our probabilistic Event-B models in terms of Markov Chains. To illustrate our proposition of probabilistic Event-B, we consider a case study of a landing gear system [7] by giving a specification of this system using probabilistic Event-B.

Outline. The paper is structured as follows. Section 2 recalls the necessary basis. In Section 3, we start by introducing the syntax of our fully probabilistic extension of

Event-B and then present the semantics of a fully probabilistic Event-B model in terms of Probabilistic Labelled Transition Systems. We also present the new proof obligations specific to our proposition and we study the required modifications on the standard proof obligations. An example of a probabilistic Event-B model is given and commented in Section 4. Section 5 concludes and presents hints for future work.

2 Background

We now briefly introduce the basic concepts used through this paper.

2.1 Event-B

Event-B [1] is a formal method used for the development of complex systems. Systems are described in Event-B by means of models. To facilitate the notation, we assume in the rest of the paper that an Event-B model is expressed by a tuple $M = (\bar{v}, I(\bar{v}), Evts, Init)$ where $\bar{v} = \{v_1, v_2, \ldots, v_n\}$ is a set of variables, $I(\bar{v})$ is the invariant, $Evts$ is a set of events and $Init \in Evts$ is the initialisation event. The invariant $I(\bar{v})$ is a conjunction of predicates over the variables of the system specifying correctness properties that must always hold.

Events. An event has the following form:

event e **any** $\bar{t}$ **where** G($\bar{t}$,$\bar{v}$) **then** S($\bar{t}$,$\bar{v}$) **end**

where e is the name of the event, $\bar{t} = \{t_1, t_2, ..., t_n\}$ represents a set of parameters of the event, $G(\bar{t}, \bar{v})$ is the guard and $S(\bar{t}, \bar{v})$ is the action of the event. An event is *enabled* only if the guard $G(\bar{t}, \bar{v})$ is satisfied. Parameters and guards are optional. The action $S(\bar{t}, \bar{v})$ of an event may contain several assignments that are executed in parallel. An assignment can be expressed in one of the following forms.

- **Deterministic assignment**: $x := E(\bar{t}, \bar{v})$ where x denotes a variable and $E(\bar{t}, \bar{v})$ denotes a set theoretic expression.
- **Predicate (non-deterministic) assignment**: $x :| Q(\bar{t}, \bar{v}, x')$ where x denotes a variable and $Q(\bar{t}, \bar{v}, x')$ represents a predicate that describes the relationship between the values of variables before ($\bar{v}$) and after ($\bar{v'}$) the occurence of the assignment; x non-deterministically takes a new value x' such that the predicate $Q(\bar{t}, \bar{v}, x')$ is satisfied.
- **Enumerated (non-deterministic) assignment**: $x :\in \{E_1(\bar{t}, \bar{v}), \ldots, E_n(\bar{t}, \bar{v})\}$, where x denotes a variable and $\{E_1(\bar{t}, \bar{v}), \ldots,$ and $E_n(\bar{t}, \bar{v})\}$ some set-theoretic expressions; x non-deterministically takes a value from the set $\{E_1(\bar{t}, \bar{v}), \ldots, E_n(\bar{t}, \bar{v})\}$.

Most of the time, when the set of values such that $Q(\bar{t}, \bar{v}, x')$ is satisfied is finite, predicate (non-deterministic) assignment can be equivalently reformulated in terms of enumerated (non-deterministic) assignment. In the rest of the paper, we thus assume that no predicate (non-deterministic) assignment is present in our models. We also assume that the initialisation event is always deterministic. We denote the guard of an event e by *grd(e)*.

Proof obligations. The consistency of an Event-B model is ensured by means of proof obligations (POs) which must be discharged. Formal definitions of all the standard POs are given in [1].
The most important PO is (event/INV) for *invariant preservation* which is expressed for an event e with guard $G(\bar{t},\bar{v})$ and action $S(\bar{t},\bar{v})$ by:

$I(\bar{v}) \wedge G(\bar{t},\bar{v}) \wedge S(\bar{t},\bar{v},\bar{v}') \vdash I(\bar{v}')$	(event/INV)

$S(\bar{t},\bar{v},\bar{v}')$ is a predicate that describes the relationship between the values of variables before ($\bar{v}$) and after ($\bar{v}'$) the occurrence of the event e. It is automatically derived from $S(\bar{t},\bar{v})$. This PO states that the invariant still holds after the execution of each event in the Event-B model M.

LTS semantics. In [6], the authors express the semantics of an Event-B model M=($\bar{v}$, $I(\bar{v})$, $Evts$, $Init$) in terms of a Labelled Transition System (LTS) $\mathcal{M}$=(S, s_0, AP, L, $Acts$, T) where S is a set of states, each state in S being uniquely identified by its label; $Acts$ is the set of actions (event names); $s_0 \in S$ is the initial state obtained by executing the $Init$ event; AP is the set of atomic propositions: a set of predicates that correspond to the valuations of $\bar{v}$ and satisfy the invariant $I(\bar{v})$; L: $S \to AP$ is a labelling function that provides the valuations of the variables $\bar{v}$ in a given state; and $T \subseteq S \times Acts \times S$ is the transition relation corresponding to the actions of the events of M.
Given an expression $E(\bar{v})$ over variables in $\bar{v}$ and a valuation $\bar{\mathcal{V}}$ of these variables, we write $[\bar{\mathcal{V}}]E(\bar{v})$ for the evaluation of $E(\bar{v})$ in the context of $\bar{\mathcal{V}}$. When it is clear from the context, we write $[\bar{\mathcal{V}}]E$ instead of $[\bar{\mathcal{V}}]E(\bar{v})$. Given a variable $x \in \bar{v}$, we also write $[\bar{\mathcal{V}}]x$ for the current value of x in $\bar{\mathcal{V}}$. For an event e in $Evts$, we write $\mathcal{E}_e(x)$ for the set of all expressions that can be assigned to x by the action of e. We say that an event $e \in Evts$ is enabled in a state s of $\mathcal{M}$ if and only if $[L(s)]grd(e) \models true$. Given a state s of $\mathcal{M}$, we denote the set of events enabled in this state by $Acts(s) = \{e \in Evts \mid [L(s)]grd(e) \models true\}$. The destination state s' of a transition $(s,e,s') \in T$ performed by an event e must satisfy the following property:

$$\forall x \in \bar{v}, \exists E \in \mathcal{E}_e(x), [L(s')](x) = [L(s)]E \tag{1}$$

2.2 Probabilistic Labelled Transition System

In Section 3, we introduce the semantics of a probabilistic Event-B model in terms of Probabilistic Labelled Transition System (PLTS for short). A PLTS is a tuple $\mathcal{M}$=(S, P, s_0, AP, L, $Acts$) where S is a set of states, $Acts$ is a set of actions, $s_0 \in S$ is the initial state, AP is a set of atomic propositions, L: $S \to AP$ is a labelling function and $\mathbf{P}$: $S \times Acts \times S \to [0,1]$ is the transition probability function.

Definition 1 (Discrete Time Markov Chain (DTMC) [13]). *A Discrete Time Markov Chain is a PLTS where for each state $s \in S$, we have $\sum_{s' \in S, a \in Acts} P(s,a,s') = 1$.*

We prove in Section 3 that the semantics of a probabilistic Event-B model is in fact a DTMC. We now move to our extension of Event-B.

3 Probabilistic Event-B

We begin by presenting the notations for probabilistic Event-B. After that, we present the semantics in terms of PLTS and we study the new POs related to our proposition.

3.1 Notations

The typical way of defining a probabilistic Event-B model from a classical Event-B model M is to go through M and replace all occurrences of non-deterministic choices with probabilistic choices. We call this process the *probabilisation* of M. In this section, we therefore explore all the potential sources of non-determinism in Event-B and explain how to turn them into probabilistic choices. We also present some new POs specific to our extension and some modifications on the standard POs.
In Event-B, a non-deterministic choice can appear in three places:

1. **Choice of the enabled event.** When several events are enabled in the same state, the event to be executed is chosen non-deterministically.
2. **Choice of the parameter values.** When an event contains some parameters, the value taken by these parameters are chosen non-deterministically from the set of values satisfying the guard of the event.
3. **Non-deterministic assignment.** When the action of an event contains non-deterministic assignments, the value to be assigned to the concerned variable is chosen non-deterministically.

We now show how we replace each such non-deterministic choice with probabilities.

Choice of the enabled event. In order to resolve this non-deterministic choice, the most intuitive solution is to annotate each event with a probability value. However, since the set of enabled events can vary from one state to another in Event-B, this solution can be incorrect. Indeed in some states, the sum of probabilities of all enabled events might end up being above or below one. We therefore annotate each event with a *weight* instead of a probability value. The weight $W_e(\bar{v})$ of a probabilistic event e is an expression over the variables $\bar{v}$ of the Event-B model; this expression is evaluated in each state according to the current values of the variables.
In a given state, the probability of each enabled event is then computed as the ratio of the value of its weight in this state against the total value of the weights of all enabled events in this state. In order to simplify notations, we impose that the weight of each probabilistic event must be a natural. A probabilistic event is allowed to be executed only if *i)* its guards is fulfilled and *ii)* its weight is strictly greater than 0.

Choice of the parameter values. For a probabilistic event e with a set of parameters $\bar{t}$, we replace the non-deterministic choice of the values of the parameters in $\bar{t}$ by a uniform choice over the set of combination of all potential values of parameters in $\bar{t}$. More precisely, each parameter $t_i \in \bar{t}$ takes its value from a set T_i. This set T_i must be finite and not empty. Then, the parameters values are chosen non-deterministically

from the set $T_1 \times ... \times T_n$ in such a way that the guard of the event is fulfilled. In a state s, we denote by T_{v_s} the set of combinations of the different parameters values that make the guard of the event fulfilled in s. Formally, we have T_{v_s}=$\{\bar{t_v} = \{t_{1_v}, t_{2_v}, ..., t_{n_v}\} \in T_1 \times ... \times T_n \mid [L(s)]G(\bar{t_v}, \bar{v}) \models true\}$. We replace the non-deterministic choice of the parameter values by an uniform choice over the set T_{v_s}. We associate a probability function $P_{T_{v_s}}$ such that for each combination $\bar{t_v} \in T_{v_s}$, we have $P_{T_{v_s}}(\bar{t_v}) = \frac{1}{card(T_{v_s})}$.

Non-deterministic assignment. Recall that we limit ourselves to enumerated (non-deterministic) assignments which are written:

$$\boxed{x :\in \{E_1(\bar{t}, \bar{v}), \ldots, E_m(\bar{t}, \bar{v})\}}$$

We propose to replace this assignment by a probabilistic assignment written as follows.

$$\boxed{x := E_1(\bar{t}, \bar{v})@_{p_1} \oplus \cdots \oplus E_m(\bar{t}, \bar{v})@_{p_m}}$$

If $m > 1$, then this new assignment assigns to the variable x an expression E_i with a probability p_i. The probability of choosing the expression E_i among all the others expressions is denoted by $P_x(E_i)$= p_i.
For each E_i, the probability $P_x(E_i)$ must be strictly greater than 0 and smaller or equal to 1. If $m = 1$, then the assignment is deterministic and written $x := E_1(\bar{t}, \bar{v})$. In this case, we have $P_x(E_1) = 1$. We note that the sum of the probabilities of the expressions that can be assigned to the variable x must be equal to 1.

3.2 Probabilistic Event-B model

The general form of a *probabilistic* event is then:

$$\boxed{\textbf{event } e \textbf{ weight } W_e(\bar{v}) \textbf{ any } \bar{t} \textbf{ where } G(\bar{t}, \bar{v}) \textbf{ then } S(\bar{t}, \bar{v}) \textbf{ end}}$$

where $S(\bar{t}, \bar{v})$ contains several probabilistic or deterministic assignments executed in parallel. Remark that a probabilistic event can be without parameters or a guard and its action must contain only probabilistic or deterministic assignments. We also note that, like in the standard Event-B, the initialisation event must be deterministic.

We present below the formal definition of a probabilistic Event-B model.

Definition 2 (Probabilistic Event-B Model). *A probabilistic Event-B model is a tuple $M = (\bar{v}, I(\bar{v}), PEvts, Init)$ where $\bar{v} = \{v_1, v_2, ..., v_n\}$ is a set of variables, $I(\bar{v})$ is the invariant, $PEvts$ is a set of probabilistic events and $Init \in PEvts$ is the initialisation event.*

Given a probabilistic event e, we write $Var(e)$ for the set of variables in $\bar{v}$ that are modified by the action of e, i.e all the variables that appear on the left side of an assignment in $S(\bar{t}, \bar{v})$. For a variable x in $Var(e)$ and an expression $E \in \mathcal{E}_e(x)$, we write $P_x^e(E)$ for the probability that the action of the event e assigns to the variable x the expression E.

DTMC semantics. The semantics of a probabilistic Event-B model $M = (\bar{v}, I(\bar{v}), PEvts, Init)$ is a PLTS where the states, labels, actions, atomic propositions and initial state are similarly obtained as for the standard LTS semantics of Event-B. Only the transitions are equipped with probabilities.
Let $e \in PEvts$ be a probabilistic event, $x \in \bar{v}$ be a variable and s, s' be two states of the corresponding PLTS such that (s,e,s') is a transition. Let $\bar{t_v} = \{t_{1_v}, t_{2_v}, \ldots, t_{n_v}\}$ be a valuation of the parameter values associated to the event e. We write $\mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}$ for the set of expressions in $\mathcal{E}_e(x)$ such that their evaluation in the state s with parameter values $\bar{t_v} = \{t_{1_v}, t_{2_v}, ..., t_{n_v}\}$ returns the value of x in the state s'. Formally,

$$\mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}} = \{E \in \mathcal{E}_e(x) \mid [L(s')]x = [L(s) \cup \bar{t_v}](E(\bar{t},\bar{v}))\}$$

If e is not equipped with parameters, then this subset is written $\mathcal{E}_e(x)|^{s'}_{s}$.
The probability of a transition (s,e,s') is then equal to the product of **(1)** the probability that the event e is chosen from the set of enabled events in state s, **(2)** the probability of choosing the parameter values $\bar{t_v}$, and **(3)** the overall probability that each modified variable is assigned the value given in $[L(s')]$. Formally, the semantics of M is defined as follows.

Definition 3 (Probabilistic Event-B Semantics).
The semantics of a probabilistic Event-B model M=($\bar{v}$, $I(\bar{v})$, PEvts, Init) is a PLTS $[\![M]\!] = (S, s_0, AP, L, Acts, P)$ *where:*

- *S is the set of states. Each state is uniquely identified by its label.*
- *$s_0 \in S$ is the initial state. This state is obtained after the execution of the Init event.*
- *AP represents the valuations of all variables that satisfy the invariant of the model:* $AP = \{P \in \prod_{x \in \bar{v}} Dom(x) \mid P \models I(\bar{v})\}$.
- *$L : S \to AP$ is a labelling function. It assigns to each state the corresponding valuation of the variables.*
- *Acts is the alphabet of actions (event names)*
- *P: $S \times Acts \times S \to$ [0,1] is the transition probability function such that for a given state s, $\forall\, e, s' \in Acts \times S$ we have $P(s,e,s') = 0$ if $Acts(s) = \emptyset$ or $\exists x \in X \backslash \{Var(e)\}$ st $[s]x \neq [s']x$ and otherwise*

$$P(s,e,s') = \underbrace{\frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})}}_{(1)} \times \sum_{\bar{t_v} \in T_{v_s}} \Big(\underbrace{P_{T_{v_s}}(\bar{t_v})}_{(2)} \times \underbrace{\prod_{x \in Var(e)} \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_x(E)}_{(3)} \Big)$$

Depending on the form of the event e, this expression can be simplified in several ways:

- If e has no parameter and its action is deterministic then:

$$P(s,e,s') = \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})}$$

- If e has some parameters and its action is deterministic then:

$$P(s,e,s') = \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})} \times \sum_{\bar{t_v} \in T_{v_s}} P_{T_{v_s}}(\bar{t_v})$$

– If e has no parameter and its action is probabilistic then:

$$P(s,e,s') = \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)}[s]W_{e'}(\bar{v})} \times \prod_{x \in Var(e)} \left(\sum_{\{E \in \mathcal{E}_e(x) | \, s'_s\}} P_x^e(E) \right)$$

Lemma 1. *Given a probabilistic Event-B model M, the semantics $[\![M]\!]$ of M is a DTMC.*

Proof. We prove that $[\![M]\!]$ is a *DTMC* by showing that the sum of the outgoing transitions from each state s in $[\![M]\!]$ is equal to one. The full proof can be found in the appendix.

After presenting the semantics of a probabilistic Event-B model, we present in what follows some new POs related to our extension.

3.3 Proof Obligations

Event-B and its dedicated tool Rodin use POs to validate Event-B models. In the long term, we will have to study the associated POs for probabilistic Event-B reasoning and derive the required modifications on the standard Event-B POs. For now, we propose some new POs specific to our extension.
The first PO concerns the weight of a probabilistic event. To facilitate notations, this PO ensures that under the invariant and the guard of a probabilistic event, its weight must be of natural type.

$I(\bar{v}) \wedge G(\bar{t},\bar{v}) \vdash W_e(\bar{v}) \in \mathbb{N}$	(event/WEIGHT)

The second new PO concerns well-definedness of a probabilistic assignment, it ensures that the probability of assigning an expression E_i to a variable x by a probabilistic assignment is a correct probability value, i.e. strictly greater than 0 and smaller or equal to 1:

$\vdash 0 < P_x(E_i) \leq 1$	(event/assign/pWD1)

The third new PO is also about well-definedness of a probabilistic assignment, it ensures that the sum of probabilities of expressions that can be assigned to a variable x is equal to 1.

$\vdash \sum_{i=1}^{m} P_x(E_i) = 1$	(event/assign/pWD2)

Our extension to Event-B implies some modifications on the standard POs. In fact, in the standard Event-B, an event is enabled only if its guard is fulfilled. For a probabilistic event, it occurs only if its guard is fulfilled *and* in addition, its weight is strictly greater than 0. As a consequence, some standard POs must be modified by adding a predicate to the guard that indicates that the weight of a probabilistic event must be strictly greater than 0. As an example, the *invariant preservation* PO (event/INV) will be expressed by:

$I(\bar{v}) \wedge G(\bar{t},\bar{v}) \wedge \mathbf{W_e}(\bar{\mathbf{v}}) > \mathbf{0} \wedge S(\bar{t},\bar{v},\bar{v}') \vdash I(\bar{v}')$	(event/pINV)

In future work, we continue our investigation in order to derive the modifications or the new POs specific to our extension.

4 Example: Landing Gear System

We consider the case study of the Landing Gear System [7] to demonstrate the usefulness of *Probabilistic Event-B* in a practical problem. We first give an overview of the case study. Then, we propose a first non-deterministic specification of the system in Event-B and we show its limitations before deriving a probabilistic specification to demonstrate the interest of our proposition.

4.1 Overview of the Landing Gear System

The landing gear system of an aircraft is in charge of manoeuvring landing gears sets, i.e gears and associated doors. We only consider the basic behaviour of this landing gear system. To command the retraction and outgoing of gears, an "up/down" handle is provided to the pilot. When the handle is switched to "up", the retraction sequence is performed. When the handle is switched to "down", the outgoing sequence is executed. The outgoing sequence consists in *i)* opening the doors, *ii)* extending the gears and *iii)* closing the doors. The retraction sequence consists in *i)* opening the doors, *ii)* retracting the gears and *iii)* closing the doors.
Some requirements constrain the model:

- R1 The pilot cannot command the handle more than a fixed number of times before one of the sequences begins;
- R2 Each consecutive use of the pilot command must decrease the priority of using it again and increase the priority of starting the outgoing/retraction sequence instead;
- R3 Failures can occur: external measures fix to 10% the risk that gears or doors do not react correctly to their command.

4.2 Non-deterministic Event-B model

The model "landing_gear" given in Fig. 1 presents an Event-B specification of the Landing Gear System. We consider four variables $handle$, $door$, $gear$ and cmd constituting the state of the modelled system.
The event *pcmd* models the pilot command, i.e. switching the handle to "up" or "down". The variable *cmd* counts the number of times this event is triggered consecutively. The guard of this event limits its enabling ($cmd \leq F\_CMD$) to take into account R1, but we are not able to consider the requirement R2 about priorities.
The event *extend* models the extending of the gears; it is enabled when the handle is down, the doors open and the gears retracted; in order to take into account possible failures, we use a non-deterministic assignment $gear :\in \{extended, retracted\}$ saying that the gears are extended (attempted behaviour) or remain retracted (failure). We cannot be more precise about the measured risk of failure as expressed in R3.
The event *retract* models the gears retraction whereas the events *open* and *close* model the opening or closing of the doors respectively: they are similarly specified as the event *extend*. When one of the event of the sequences is triggered, then the counter *cmd* is reset.

model
landing_gear
constants
F_CMD
axioms
F_CMD $\in \mathbb{N}_1$
variables
handle
door
gear
cmd
invariants
handle $\in$ {up, down}
door $\in$ {open, closed}
gear $\in$ {extended, retracted }
cmd $\in \mathbb{N}$
events
event initialisation
then
handle := up
door := closed
gear := retracted
cmd := 0
end
event pcmd
any cc
where
cc $\in$ {up,down} $\wedge$ cmd $\leq$ F_CMD
then
handle := cc
cmd := cmd+1
end

event extend
when
handle=down $\wedge$ door=open $\wedge$ gear=retracted
then
gear :$\in$ {extended, retracted }
cmd := 0
end
event retract
when
handle= up $\wedge$ door=open $\wedge$ gear=extended
then
gear :$\in$ {extended, retracted }
cmd := 0
end
event open
when
door=closed $\wedge$
((handle=down$\wedge$gear=retracted)
$\vee$ (handle=up$\wedge$gear=extended))
then
door :$\in$ {open,closed}
cmd := 0
end
event close
when
door=open $\wedge$
((handle=down$\wedge$gear=extended)
$\vee$ (handle=up $\wedge$ gear= retracted))
then
door :$\in$ {closed,open}
cmd := 0
end

Fig. 1: Event-B model of the Landing Gear System

4.3 Probabilistic Event-B model

The model "proba_landing_gear" given in Fig. 2 presents a probabilistic version of the "landing_gear" model.

To resolve the non-determinism between events that are enabled in the same states, we annotate the events by weights. To express that the more the pilot uses the "up/down" handle, the less this command will be taken into account – with a limit of *F_CMD* times – we annotate the event *pcmd* by the weight $F\_CMD - cmd$ and the events *open*, *close*, *extend* and *retract* by $F\_CMD + cmd$. The weight of the event *pcmd* therefore decreases each consecutive time it is executed whereas the weight of the other events increases. As a consequence, the more this event is activated in a consecutive sequence, the higher the chance that this sequence will end and that another event *open*, *close*, *extend* or *retract* will be activated instead (Requirement R2).

model
proba_landing_gear
probabilises
landing_gear
events
event initialisation
then
handle := up
door := closed
gear := retracted
cmd := 0
end
event pcmd
probabilises pcmd
weight
F_CMD − cmd
any
cc
where
cc ∈ {up,down} ∧ cmd ≤ F_CMD
then
handle := cc
cmd := cmd + 1
end
event extend
probabilises extend
weight
F_CMD + cmd
when
handle=down ∧ door=open ∧ gear=retracted
then
gear:= extended@9/10 ⊕ retracted@1/10
cmd := 0
end

event retract
probabilises retract
weight
F_CMD + cmd
when
handle= up ∧ door=open ∧ gear=extended
then
gear:= extended@1/10 ⊕ retracted@9/10
cmd := 0
end
event open
probabilises open
weight
F_CMD + cmd
when
door=closed
∧ ((handle=down∧gear=retracted)
∨ (handle=up∧gear=extended))
then
door:= open@9/10 ⊕ closed@1/10
cmd := 0
end
event close
probabilises close
weight
F_CMD + cmd
when
door=open
∧ ((handle=down∧gear=extended)
∨ (handle=up ∧ gear= retracted))
then
door:= open@1/10 ⊕ closed@9/10
cmd := 0
end

Fig. 2: Probabilistic model of the Landing Gear System

Failures are then described by means of probabilistic assignments. The event *extend* assigns probabilistically to the variable *gear* the value *extended* with probability $\frac{9}{10}$ and the value *retracted* with probability $\frac{1}{10}$ (probability of failure) as required in R3. Similarly, we model failures in the events *retract*, *open* and *close*.

Proof Obligations. The consistency of the given probabilistic Event-B model requires to discharge some POs. The POs (event/assign/pWD1) and (event/assign/pWD2) obviously holds for all the events having probabilistic assignments:

$$\vdash 0 < \frac{9}{10} \leq 1, \quad \vdash 0 < \frac{1}{10} \leq 1 \text{ and } \vdash \frac{9}{10} + \frac{1}{10} = 1$$

We have to demonstrate that the PO (event/WEIGHT) holds for the event *pcmd*. As hypotheses, we have $F\_CMD \in \mathbb{N}_1$ and $cmd \in \mathbb{N}$ (invariant), and $cmd \leq F\_CMD$ (guard). We show that the goal $F\_CMD - cmd \in \mathbb{N}$ holds.

The PO (event/WEIGHT) for the other events is similarly demonstrated.

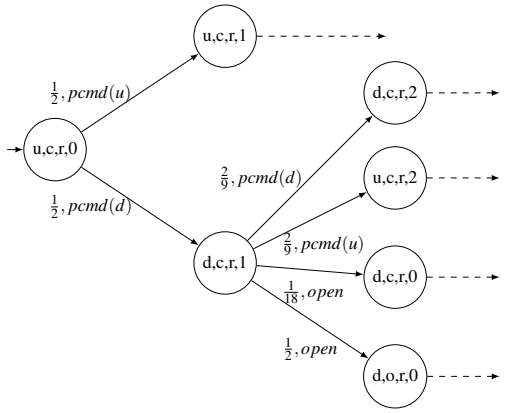

Fig. 3: DTMC part of the probabilistic LGS with $F\_CMD = 9$

DTMC semantics. Fig. 3 presents a part of the DTMC corresponding to "proba_ Landing_gear". We only present this part of DTMC to show the semantics of our model and to prove the correctness of our approach, this DTMC will not be used within the design process in probabilistic Event-B. The states of this DTMC correspond to the valuations of the variables *handle*, *door*, *gear* and *cmd*. The transitions between the states correspond to the possible occurrence of the events, labelled with their probability value. We have to fix a value for the *F_CMD* constant, 9 for the example below.

In the state $(d,c,r,1)$ only two events can be enabled: *open* and *pcmd*[1]. The event *open* leads to the state $(d,o,r,0)$ with a probability $\frac{1}{2} = \frac{10}{10+8} \times \frac{9}{10}$, where $\frac{10}{10+8}$ corresponds to the probability of choosing the event *open* rather than the event *pcmd* while $\frac{9}{10}$ corresponds to the probability of assigning the value *open* rather than the value *closed* to the variable *door*; Similarly, the event *open* leads to the state $(d,c,r,0)$ with a probability $\frac{1}{18} = \frac{10}{10+8} \times \frac{1}{10}$. The event *pcmd* leads to $(d,c,r,2)$ or $(u,c,r,2)$ with the same probability $\frac{2}{9} = \frac{8}{10+8} \times \frac{1}{2}$ where $\frac{8}{10+8}$ corresponds to the probability of choosing *pcmd* rather than *open* while $\frac{1}{2}$ is the probability of choosing *up* (respectively *down*) as value for the event parameter *cc*. Note that $\frac{1}{2} + \frac{1}{18} + \frac{2}{9} + \frac{2}{9} = 1$, i.e. the sum of probabilities of the outgoing transitions from the state $(d,c,r,1)$ is equal to one, as expected.

The probabilities of all the transitions in the DTMC could be similarly computed.

[1] In the state $(d,c,r,1)$, the weight of *open* is 10 whereas the weight of *pcmd* is 8.

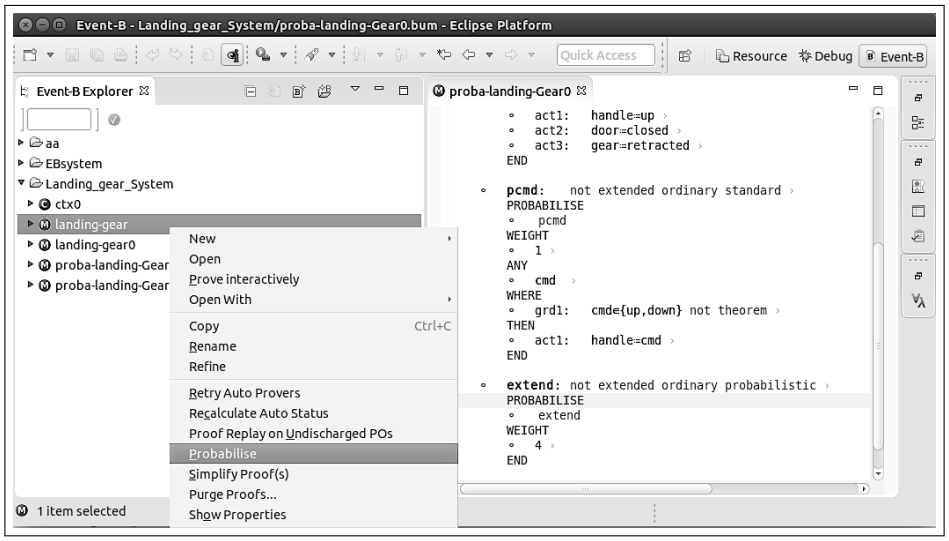

Fig. 4: Probabilistic plugin to the Rodin platform

5 Conclusion

In this paper, we present an extension to Event-B that allows us to specify fully probabilistic systems. Previous works combining probabilities and Event-B [14,11,16,17] have focused in refining non-deterministic assignments into probabilistic ones while leaving other sources of non-determinism untouched. This work extends these previous works by replacing all the non-deterministic choices by probabilistic ones. More precisely, we extend the syntax of Event-B to add probabilistic informations and we derive some new POs specific to our extension: each event is annotated with a positive weight, parameters values are uniformly chosen and non-deterministic assignments are replaced with quantitative probabilistic assignments as in [16,17]. Then, we show that the semantics of our fully probabilistic Event-B models are, as expected, Discrete Time Markov Chains.

We have started the development of a new plugin to the Rodin Platform, this plugin allows the specification and verification of probabilistic Event-B models. The current features are listed in Tab. 1 where ✓ denotes the supported functionalities and ∼ the functionalities currently under development. This plugin allows the specification of fully probabilistic Event-B models and the *probabilisation* of an Event-B model as shown in Fig. 4.

As argued in the introduction, our aim in the long term is to develop a probabilistic extension of Event-B where the developer can choose at his convenience where to refine non-deterministic choices with probabilities and where to keep non-deterministic choices intact. This paper is a first step towards this long term goal.

Annotating events with weights	✓
Checking new PO (event/WEIGHT)	✓
Introducing new probabilistic assignments	✓
Checking new PO (event/assign/pWD)	∼
Checking new PO (event/assign/pWD2)	✓
Updating standard POs	∼
Implementing the *probabilisation* process	✓

Table 1: Probabilistic Plugin features

About refinement. As the development in Event-B is intrinsically based on a refinement process, we plan on studying the refinement of fully probabilistic Event-B models. In this paper, we have only considered a (kind of) refinement step consisting to move from a purely non-deterministic Event-B model to a purely probabilistic Event-B model (the probabilisation process). We obviously intend to study and develop refinement *between* probabilistic Event-B models.
In Event-B, introducing new events by refinement implies the proof of the convergence of these new events (i.e. a decreasing of a given variant). Probabilities can facilitate the proof as shown in [11].

About properties verification. Most of the properties of interest that are verified in classical Event-B are safety related. They are most of the time expressed by means of invariants and discharged as POs. In addition, a critical system must also satisfy some liveness properties.
In [5], Abrial and *al.* have proposed a way to express some dynamic constraints in B. In [10], the authors have extended this work and have proposed a solution to verify some temporal properties expressed in LTL on Event-B models. In this direction, the work in [9] adresses the verification of PLTL properties by model checking. In [12], the authors have proposed a set of proof rules for reasoning about some liveness properties (existence, progress, persistence). In [4], the authors have expressed and proved reachability within the refinement process in Event-B.
Based on these works, we will investigate in the future how we can verify some properties expressed in some probabilistic logic like PLTL or PCTL on probabilistic Event-B models.

In parallel to the reporting work, we intend to improve our probabilistic plugin by implementing new features such as probabilistic properties and refinement of probabilistic Event-B models.

References

1. J.-R. Abrial. *Modeling in Event-B: system and software engineering*. Cambridge University Press, 2010.
2. J.-R. Abrial, M. Butler, S. Hallerstede, T. S. Hoang, F. Mehta, and L. Voisin. Rodin: an open toolset for modelling and reasoning in event-b. *International journal on software tools for technology transfer*, 12(6):447–466, 2010.
3. J.-R. Abrial, D. Cansell, and D. Méry. A mechanically proved and incremental development of ieee 1394 tree identify protocol. *Formal aspects of computing*, 14(3):215–227, 2003.
4. J.-R. Abrial, D. Cansell, and D. Méry. Refinement and reachability in event-b. In H. Treharne, S. King, M. Henson, and S. Schneider, editors, *ZB 2005: Formal Specification and Development in Z and B*, volume 3455 of *Lecture Notes in Computer Science*, pages 222–241. Springer Berlin Heidelberg, 2005.
5. J.-R. Abrial and L. Mussat. Introducing dynamic constraints in b. In D. Bert, editor, *B98: Recent Advances in the Development and Use of the B Method*, volume 1393 of *Lecture Notes in Computer Science*, pages 83–128. Springer Berlin Heidelberg, 1998.
6. D. Bert and F. Cave. Construction of finite labelled transition systems from b abstract systems. In *Integrated Formal Methods*, volume 1945 of *LNCS*, pages 235–254. Springer, 2000.

7. F. Boniol and V. Wiels. The landing gear system case study. In *ABZ 2014: The Landing Gear Case Study*, pages 1–18. Springer, 2014.
8. W. W. Chu and C.-M. Sit. Estimating task response time with contentions for real-time distributed systems. In *Real-Time Systems Symposium, 1988., Proceedings.*, pages 272–281. IEEE, 1988.
9. C. Darlot, J. Julliand, and O. Kouchnarenko. Refinement preserves pltl properties. In D. Bert, J. Bowen, S. King, and M. Waldén, editors, *ZB 2003: Formal Specification and Development in Z and B*, volume 2651 of *Lecture Notes in Computer Science*, pages 408–420. Springer Berlin Heidelberg, 2003.
10. J. Groslambert. Verification of ltl on b event systems. In J. Julliand and O. Kouchnarenko, editors, *B 2007: Formal Specification and Development in B*, volume 4355 of *Lecture Notes in Computer Science*, pages 109–124. Springer Berlin Heidelberg, 2006.
11. S. Hallerstede and T. S. Hoang. Qualitative probabilistic modelling in event-b. In *Integrated Formal Methods*, pages 293–312. Springer, 2007.
12. T. Hoang and J.-R. Abrial. Reasoning about liveness properties in event-b. In S. Qin and Z. Qiu, editors, *Formal Methods and Software Engineering*, volume 6991 of *Lecture Notes in Computer Science*, pages 456–471. Springer Berlin Heidelberg, 2011.
13. J. G. Kemeny and J. L. Snell. *Finite markov chains*, volume 356. van Nostrand Princeton, NJ, 1960.
14. C. Morgan, T. S. Hoang, and J.-R. Abrial. The challenge of probabilistic event b—extended abstract—. In *ZB 2005: Formal Specification and Development in Z and B*, pages 162–171. Springer, 2005.
15. R. Motwani and P. Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010.
16. A. Tarasyuk, E. Troubitsyna, and L. Laibinis. Reliability assessment in event-b development. *NODES 09*, page 11, 2009.
17. A. Tarasyuk, E. Troubitsyna, and L. Laibinis. Towards probabilistic modelling in event-b. In *Integrated Formal Methods*, pages 275–289. Springer, 2010.
18. A. Tarasyuk, E. Troubitsyna, and L. Laibinis. Integrating stochastic reasoning into event-b development. *Formal Aspects of Computing*, 27(1):53–77, 2015.
19. K. S. Trivedi, S. Ramani, and R. Fricks. Recent advances in modeling response-time distributions in real-time systems. *Proceedings of the IEEE*, 91(7):1023–1037, 2003.
20. A. Villemeur. *Reliability, Availability, Maintainability and Safety Assessment, Assessment, Hardware, Software and Human Factors*, volume 2. Wiley, 1992.
21. E. Yilmaz. *Tool support for qualitative reasoning in Event-B*. PhD thesis, Master Thesis ETH Zürich, 2010, 2010.

Proof of Lemma 1

Given a probabilistic Event-B model M, the semantics $[\![M]\!]$ of M is a *DTMC*.

Proof. We must prove that for each state s in $[\![M]\!]$, the sum of probabilities of the outgoing transitions from s is equal to one. Let M be a probabilistic Event-B model, $\bar{v} = (x_1, x_2, ..., x_n)$ the set of variables of M and $s \in S$ a state of $[\![M]\!]$. We assume that each variable x_i in $\bar{v}$ take its value from a set X_i.
Recall that the probability of a transition (s,e,s') is 0 if $Acts(s) = \emptyset$ or $\exists x \in \bar{v} \backslash \{Var(e)\}$ st $[s]x \neq [s']x$ and otherwise:

$$P(s,e,s') = \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)}[s]W_{e'}(\bar{v})} \times \sum_{\bar{t_v} \in T_{v_s}} \left[P_{T_{v_s}}(\bar{t_v}) \times \prod_{x \in Var(e)} \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_x(E)\right]$$

We must therefore show that $\sum_{e \in Acts(s)} \sum_{s' \in S} P(s,e,s') = 1$.

$$\sum_{s' \in S, e \in Acts(s)} P(s,e,s') = \sum_{e \in Acts(s)} \sum_{s' \in S} \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)}[s]W_{e'}(\bar{v})} \times \sum_{\bar{t_v} \in T_{v_s}} \left[P_{T_{v_s}}(\bar{t_v}) \times \prod_{x \in Var(e)} \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_x(E)\right]$$

$$\sum_{s' \in S, e \in Acts(s)} P(s,e,s') = \sum_{e \in Acts(s)} \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)}[s]W_{e'}(\bar{v})} \times \sum_{\bar{t_v} \in T_{v_s}} \left[P_{T_{v_s}}(\bar{t_v}) \times \sum_{s' \in S} \prod_{x \in Var(e)} \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_x(E)\right]$$

Let $S_1 = \{s' \in S | \forall x \in \bar{v} \backslash Var(e).[s]x = [s']x\}$.

$$\sum_{s' \in S, e \in Acts(s)} P(s,e,s') = \sum_{e \in Acts(s)} \frac{[s]W_e(\bar{v})}{\sum_{e' \in Acts(s)}[s]W_{e'}(\bar{v})} \times \sum_{\bar{t_v} \in T_{v_s}} \left[P_{T_{v_s}}(\bar{t_v}) \times \sum_{s' \in S_1} \prod_{x \in Var(e)} \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_x(E)\right]$$

First, we remark that $\mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}$ does not really depend on s' but only depends on $v'_x = [s']x$ (As s' corresponds to the valuations of the variables x_i in the state s').
Given $x \in \bar{v}$ and $v'_x \in X$, we therefore write $F^{s,\bar{t_v},e}_x(v'_x) = \sum_{E \in \mathcal{E}_e(x)|^{s'}_{s,\bar{t_v}}} P^e_{x_i}(E)$ if $x \in Var(e)$.
For $\bar{v} = \{x_1, ..., x_n\}$, we have $S_1 = \{(v_{x_1}, ..., v_{x_n}) | v_{x_i} = [s]x_i$ if $x_i \in Var(e)$ and $v_{x_i} \in X_i$ otherwise$\}$.
We assume that $Var(\bar{e}) = \{x_1, ..., x_k\}$ with $k \leq n$,
Then for all expression α with $\alpha = [v_{x_{k+1}} \leftarrow [s]x_{k+1}, ..., v_{x_n} \leftarrow [s]x_n]$ we have:

$$\sum_{s' \in S_1} \alpha = \sum_{v_{x_1} \in X_1} \left(\sum_{v_{x_2} \in X_2} (... \sum_{v_{x_k} \in X_k} \alpha)\right)$$

As a consequence, we have:

$$\sum_{s' \in S_1} \prod_{x_i \in Var(e)} F^{s,\bar{t_v},e}_{x_i}([s']x_i) = \sum_{v_{x_1} \in X_1} \left(\sum_{v_{x_2} \in X_2} (... \sum_{v_{x_k} \in X_k} (\prod_{i=1}^{k} F^{s,\bar{t_v},e}_{x_i}(v_{x_i})))\right)$$

$$= \sum_{v_{x_1} \in X_1} (\sum_{v_{x_2} \in X_2} (... \sum_{v_{x_k} \in X_k} (F^{s,\bar{t_v},e}_{x_1}(v_{x_1}).F^{s,\bar{t_v},e}_{x_2}(v_{x_2})...F^{s,\bar{t_v},e}_{x_k}(v_{x_k}))))$$
$$= [\sum_{v_{x_1} \in X_1} F^{s,\bar{t_v},e}_{x_1}(v_{x_1})].[\sum_{v_{x_2} \in X_2} F^{s,\bar{t_v},e}_{x_2}(v_{x_2})]...[\sum_{v_{x_k} \in X_k} F^{s,\bar{t_v},e}_{x_k}(v_{x_k})]$$

$$= \prod_{x_i \in Var(e)} [\sum_{v_{x_i} \in X_i} F_{x_i}^{s,\bar{t_v},e}(v_{x_i})]$$

By construction,for $x_i \in Var(e)$, we have $\sum_{v_{x_i} \in X_i} F_{x_i}^{s,\bar{t_v},e}(v_{x_i}) = 1$
Therefore, $\sum_{s' \in S_1} \prod_{x \in Var(e)} F_x^{s,\bar{t_v},e}([s']x) = 1$.
As a consequence,

$$\sum_{s' \in S, e \in Acts(s)} P(s,e,s') = \sum_{e \in Acts(s)} [\frac{[s]W_e(\bar{v}).\sum_{\bar{t_v} \in T_{v_s}} P_{T_{v_s}}(\bar{t_v})}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})}]$$

$$= \frac{\sum_{e \in Acts(s)} [s]W_e(\bar{v}).\left(\sum_{\bar{t_v} \in T_{v_s}} P_{T_{v_s}}(\bar{t_v})\right))}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})}$$

By construction, we have $\sum_{\bar{t_v} \in T_{v_s}} P_{T_{v_s}}(\bar{t_v}) = 1$ and thus:

$$\sum_{s' \in S, e \in Acts(s)} P(s,e,s') = \frac{\sum_{e \in Acts(s)} [s]W_e(\bar{v})}{\sum_{e' \in Acts(s)} [s]W_{e'}(\bar{v})} = 1$$

As a conclusion, we have that $\forall s, \sum_{s' \in S, e \in Acts(s)} P(s,e,s') = 1$ and then M is a DTMC □